Amendments to the Specification

Please replace the paragraph that begins on Page 1, line 5 and carries over to Page 2, line 2, with the following marked-up replacement paragraph:

-- The present invention is related to U. S. Patent _____ (serial number ~~10/_____~~) 10/077,547), entitled "Programmatically Deriving Street Geometry from Address Data"; U. S. Patent _____ (serial number ~~10/_____~~) 10/077,079), entitled "Adapting Point Geometry for Storing Address Density"; and U. S. Patent _____ (serial number ~~10/_____~~) 10/077,146), entitled "Programmatically Calculating Paths from a Spatially-Enabled Database", each of which was filed concurrently herewith and which is hereby incorporated herein by reference. These patents are commonly assigned to the International Business Machines Corporation ("IBM"). The latter of these patents is referred to hereinafter as "the path computation invention". --

Please replace the paragraph that begins on Page 5, line 17 and carries over to Page 6, line 11, with the following marked-up replacement paragraph:

-- As one example of a spatially-enabled database, a feature known as "Spatial Extender" can be added to IBM's DB2® relational database product to provide GIS support. Spatial Extender provides support for the geometric data types shown in Fig. 1, and provides a number of built-in functions for operating on those data types. When using Spatial Extender, spatial data can be stored in columns of spatially-enabled database tables by importing the data or deriving it. The import process uses one of the WKT, WKB, or ".shp" shape formats described above as source data, and processes that data using built-in functions to convert it to geometric data. For

example, WKT format data may be imported using "geometryFromText" functions; similar functions are provided for WKB format data ("geometryFromWKB") and ".shp" shape data ("geometryFromShape"). Spatial data may be derived either by operating on existing geometric data (for example, by defining a new polygon as a function of an existing polygon) or by using a process known as "geocoding". A geocoder is ~~provider~~ provided with Spatial Extender that takes as input an address in the United States, and derives a geometric point representation. Other geocoders can be substituted to provide other types of conversions. --

Please replace the paragraph on Page 7, lines 4 - 9, with the following marked-up replacement paragraph:

-- While WKT is an open, interchangeable data format, it may be considered as a relatively "artificial" or "contrived" format for source data. That is, all geometric data that is expressed in WKT format must be specified using particular syntax conventions. To represent the point having an x-coordinate of 12 and y-coordinate of 25, commonly denoted as (12,25), for example, the following WKT syntax is used:

'point (12 25 ~~15~~)' --

Please replace the paragraph on Page 14, lines 2 - 7, with the following marked-up replacement paragraph:

-- The textual input data from which street geometry is derived may contain a number of different types of values which, taken collectively, may be referred to as "address data". The

textual data may also contain non-address-related information, such as the names of people and/or a business associated with a particular address. For example, a representative entry in the textual input data file might be as follows:

John and Mary Doe, 123 Main Street, Raleigh, NC 27613 (919) 555-1212 --

Please replace the paragraph on Page 18, lines 4 - 14, with the following marked-up replacement paragraph:

-- Each record in address table 500 has a unique index ("addr_id" in this example). In preferred embodiments, the full street address is stored in a column ("address" in this example) of the address table, in text format. A "street_id" column provides a pointer or reference which refers to a record in the street table 530. (This pointer provides a link between the address record in table 500 and the geometry data for the corresponding street. Preferably, this value is an alternate key whose value is unique in each row.) The "city", "state", and "zipcode" columns of address table 500 preferably store a textual representation of the city name, state name, and zip code associated with this address. Optionally, the key value corresponding to the values in one or more of these columns may be stored in addition to, or instead of, the textual values. Considerations in the choice of storage representation for these values ~~includes~~ include anticipated use of the data mart. --

Please replace the paragraph that begins on Page 19, line 9 and carries over to Page 20, line 2, with the following marked-up replacement paragraph:

-- Street table 530 contains street geometry data, and table 530 corresponds to street table 250 in the data mart schema representation in Fig. 2. Values in the rows of street table 530 are created while processing the input ~~table~~ file, as will be described with reference to Fig. 10. The sample values in the three rows of street table 530 represent the three sample rows of address table 500. (In an actual spatially-enabled database, address table 500 may contain many more rows than street table 530.) Each row of street table 530 begins with a key ("street_id" in this example) that refers to the street_id column of address table 500. The starting point ("start_Pt") for each street is preferably stored as a column of the street table, using an <x,y> coordinate representation of the latitude and longitude where (for purposes of the set of data in this database) this street begins. The street name is preferably stored in text form within each record (in the column "name", in this example). Each row also preferably contains an "envelope" column and a "linestring" column, where the envelope column stores a bounding box corresponding to the path taken by this street. The value of the envelope column is created in a manner that is analogous to that which has been described for the envelope column of the state table 400, by invoking the ST_Envelope function with the street's linestring as an input parameter. --

Please replace the paragraph on Page 20, lines 3 - 19, with the following marked-up replacement paragraph:

-- The last column of street table 530, designated as ~~"Point_ZM", is~~ "PointZM", is a 4-dimensional value in preferred embodiments. As discussed earlier, 3-dimensional and 4-dimensional extensions have been defined for the WKT and WKB formats, and the ~~Point_ZM~~

PointZM form <x,y,z,m> corresponds to this 4-dimensional extension. According to preferred embodiments, the values of these 4 dimensions are used in a novel way to provide a compact technique for storing information about the corresponding street. Prior art uses for these four dimensions provide a latitude, longitude, elevation/depth, and measure/distance value. (As stated earlier, values which result after applying an offset may be stored in these dimensions, rather than actual values, but that distinction is not pertinent to the present discussion.) As defined by the present invention,

- the first dimension of ~~Point_ZM~~ PointZM entries in table 530 stores a state_id value, which provides a reference to the state table (see table 400 of Fig. 4);
- the second dimension stores a city_id value, providing a reference to the city table (see table 430 of Fig. 4);
- the third dimension stores a zip_id value, providing a reference to the zip code table (see table 460 of Fig. 4); and
- the fourth dimension stores a density value, representing the density of addresses on this particular street. --

Please replace the paragraph on Page 30, lines 14 - 15, with the following marked-up replacement paragraph:

-- Block 830 stores the located city_id and state_id values into the corresponding columns of the zip code table, and Block 835 inserts the zip code value into the zipcode column. --

Please replace the paragraph on Page 32, lines 1 - 7, with the following marked-up replacement paragraph:

-- Block 905 parses the input record into street address, city, state, and zip code elements. Block 910 then obtains an (x,y) coordinate representation of this address. As stated earlier, an embodiment of the present invention may obtain this representation ~~is~~ in several alternative ways. In one approach, the (x,y) coordinates may be included in the input record. In another approach, a lookup function may be used to determine a mapping of an address to a point representing that address's geographic location. Alternative techniques for obtaining the (x,y) coordinates may be substituted without deviating from the scope of the present invention. --

Please replace the paragraph that begins on Page 35, line 13 and carries over to Page 36, line 1, with the following marked-up replacement paragraph:

-- The x-coordinate for the PointZM column entry is set to the state_id of the state in which the street corresponding to this street table row is located, the y-coordinate is set to the city_id for the city in which this street is located, and the z-coordinate is set to the zip_id corresponding to the street table row. The state_id, city_id, and zip_id values are readily available if this processing is integrated with the logic of Figs. 6 - 8, or may be determined using lookup techniques as has been described. (See the discussion of Block ~~935~~ 825, above, for example.) The m-coordinate is used as a counter, in preferred embodiments, to calculate the density of addresses occurring on this street. Thus, the m-coordinate is initialized to one during the processing of Block 1035. --

Please replace the paragraph on Page 43, lines 15 - 19, with the following marked-up replacement paragraph:

-- One or more side ~~table~~ tables, such as points of interest table 270 of Fig. 2, may also be created, if desired. As shown in Fig. 2, the sample table includes an index ("rid"), a "type" column (identifying the type of landmark represented by this row, for example), a "name" column (providing the name of the landmark), and a "phone" column (providing a phone number of the landmark). --